Topology and its Applications 46 (1992) 199-217
North-Holland

Tests for connectivity preservation for parallel reduction operators

Richard W. Hall

Electrical Engineering Department, University of Pittsburgh, Pittsburgh, PA 15261, USA

Received 14 June 1990
Revised 21 August 1991

Abstract

Hall, R.W., Tests for connectivity preservation for parallel reduction operators, Topology and its Applications 46 (1992) 199-217.

Connectivity preservation is a concern in the design of image processing algorithms for parallel reduction processes like thinning where connected components in 2-D images are reduced to medial axis approximations. Tests and proof techniques, which use local support for their computations, have been presented by Rosenfeld, Ronse and others to prove preservation of connectivity for reduction classes of thinning algorithms. In this paper the earlier Rosenfeld proof techniques are characterized as connectivity preservation tests for rectangular and hexagonal tessellations and simplified versions of these tests are developed. The later Ronse tests are characterized for rectangular and hexagonal tessellations; the Ronse test complexities are evaluated; and relevant equivalences are identified between the Rosenfeld and Ronse test forms. It is shown that forms of the Ronse tests for rectangular and hexagonal tessellations are derivable directly and simply from tests derived from the earlier Rosenfeld proof techniques. Arguments are given to establish under what operator support restrictions these tests are necessary and sufficient proofs for key connectivity preservation properties. Finally, a specific computer-based implementation of the Ronse test is described with extensions for reduction operators with larger supports and with reduction operators using subfields notions; and time performance results are given for this implementation.

Keywords: Binary digital images, parallel thinning algorithms, parallel reduction operators, tests for connectivity preservation, rectangular and hexagonal tessellations, Ronse tests, computer implementation.

AMS (MOS) Subj. Class.: Primary 54D05, 68U05, 68U10, 68R99, 52C99, 68T10.

1. Introduction

In image processing it is usually desirable to reduce the complexity of an image by determining some simpler representation for parts of the image. For example, an image may be preprocessed into a set of regions, perhaps representing objects, each of which is then reduced to a simpler representation. This representation might

Correspondence to: Professor R.W. Hall, Electrical Engineering Department, University of Pittsburgh, Pittsburgh, PA 15261, USA.

be an approximation of the medial axis [18] for each region or object when these are elongated. The process which determines these medial axis approximations is usually referred to as thinning. Thinning processes typically reduce an object by successively removing border points of the object while maintaining connectivity. An example of this process is illustrated in Fig. 1. Processes or operators which transform images only by removing object points are referred to as reduction operators. Sequential implementations of reduction operators, where only a single pixel is changing at any one time, have well-known and simply derived necessary and sufficient conditions for preserving connectivity properties [16, 20]. But, massively parallel computers are becoming more available and larger; this increases the interest in parallel implementations of image processing algorithms [3, 10, 13]. Parallel implementations of reduction operators create a more complex situation since large numbers of pixels are changing simultaneously. This complicates the proof of connectivity preservation for these parallel reduction operators. In the image processing community parallel approaches to thinning using reduction operators have received substantial attention, but the care taken to preserve or prove connectivity properties has been mixed [1, 2, 8, 12, 19, 21]. There is some need for relatively available and efficient techniques for proving these types of properties. When these proof techniques are stated as procedures or algorithms, they are

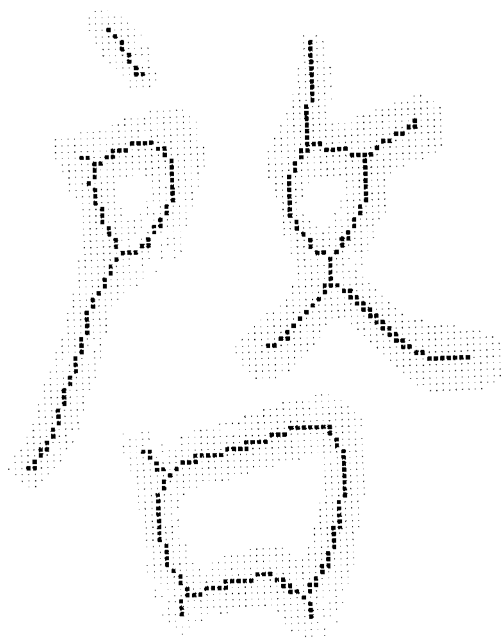

Fig. 1. Example of thinning for a Chinese character. Smaller symbols · represent pixels of the original image which have been removed by a thinning algorithm. Larger symbols ■ represent remaining pixels of the original image which form the medial axis approximation formed by the thinning algorithm. (These results are obtained using the algorithm, HSCPN, described in Section 2.)

referrred to as *connectivity preservation tests.* With more readily available proof techniques, algorithm designers can more easily prove the correctness of their image processing algorithms based on reduction operators. If one has connectivity preservation tests which can be efficiently realized (i.e., with fast execution times) in a computer program, algorithm designers can further improve the efficiency of their design process by automating the proofs of algorithms or operators under test. This need for available and efficient proofs and related connectivity preservation tests is a principal rationale for this work.

There has been a body of work over the past two decades addressing this need in rectangular tessellations. The proof techniques of Rosenfeld [17, 19] provide the rationale for proving certain key connectivity properties in thinning based on reduction operators. Rao et al. [9] gave a connectivity preservation test for a very restrictive special case among such operators. Hall [7] has determined simple sufficient conditions for a large subclass of all parallel thinning algorithms based on reduction operators. Ronse [14, 15] has presented a set of sufficient conditions which represent a particularly simple set of connectivity preservation tests for the general case of parallel thinning based on reduction operators in rectangular tesselations.

In this paper certain fundamental notation is defined in Section 2, then in Section 3 the traditional techniques of Rosenfeld [17] for proving connectivity preservation and other related connectivity preservation tests [7] in rectangular tessellations are characterized in both rectangular and hexagonal tessellations and simplified tests are demonstrated. In Section 4 the Ronse [15] connectivity preservation tests for rectangular tessellations are extended to hexagonal tessellations and efficient instances of these tests are presented. Implementation complexities for Ronse tests are then explored; and it is shown that test complexities for rectangular spaces are substantially greater than those for hexagonal spaces. It is also shown that the Ronse tests are derivable directly and simply from the earlier proof techniques and that essential elements of each test form are equivalent. These tests are sufficiency tests (i.e., satisfying the tests implies connectivity properties are satisfied); but, the *necessity* of the tests (i.e., satisfying connectivity properties implies satisfaction of the tests) has also been addressed in Section 5. Under certain restrictions on operator support size, it is shown that the connectivity preservation tests are necessary and sufficient. Finally, in Section 6 details of a computer implementation of the Ronse test are given.

2. Image spaces and parallel reduction operators

2.1. Image space definitions

Two-dimensional (2-D) image spaces are considered which are defined by rectangularly or hexagonally tessellating $\mathbb{R}^2$ with square or hexagonal cells, respectively. This discretizes $\mathbb{R}^2$ and in practice these discrete image spaces are of finite size, although they are treated as unbounded here. Each square or hexagonal cell in the tessellation is referred to as a *pixel.* For rectangular tessellations the 4-*adjacent* (8-*adjacent*) neighbors of a pixel, p, are defined as the four (eight) pixels which share a common cell edge with p (share a common edge or common vertex with p). In hexagonal tessellations the 6-*adjacent* neighbors of p are defined as the six pixels which share a common edge with p. Within a given set of pixels, X, an *i-path* from p_1 to p_n (for $i = 4, 6$, or 8) is defined as a sequence of pixels of X, $\{p_1 p_2 p_3 \ldots p_n\}$, where each p_j is i-adjacent to p_{j-1} and p_{j+1} for $j = 2$ to $n-1$. Two pixels are *i-connected* in X if they belong to an i-path in X and a set of pixels is i-connected in X if each pair of its elements is i-connected in X. An *i-connected component* of X (*i-component*) is defined as a maximal i-connected subset of X. If for a set of pixels X connectivity definitions based on i-adjacency are used, then it is said that X has been defined with *i-connectivity.* Pixel values are assigned from the binary set $\{0, 1\}$ and 1-valued pixels are termed **ones** and 0-valued pixels are termed **zeros**. The set S of **ones** is referred to as the *foreground,* its complement, the set S' of **zeros**, the *background.* In rectangular tessellations to avoid connectivity paradoxes S and S' are defined with 8-connectivity and 4-connectivity, respectively [11, 16, 18]. This is referred to as an 8-4 image space. The "dual" 4-8 case could also be used, but usually the image processing community assumes 8-connectivity for foreground regions in the image space and only the 8-4 rectangular case is considered in this paper. The hexagonal tessellation uses 6-connectivity for both foreground and background and this image space is referred to as 6-6. Variables k and m will be used exclusively to refer to the foreground and background adjacency relations, respectively. For 8-4 cases $k = 8$ and $m = 4$ and for 6-6 cases $k = m = 6$. Lower case letters will be used to denote pixels and upper case letters will be used to denote sets of pixels including paths. $N_i(p)$ refers to the set containing p and its i-adjacent neighbors; $N_i^*(p) \equiv N_i(p) - \{p\}$; $N_i(P)$ is the union of $N_i(p_j)$ for all p_j in P; and $N_i^*(P) \equiv N_i(P) - P$, all for $i = 4, 6$ or 8. In illustrations of regions of the image space $\langle s \rangle$ refers to the set of all pixels labeled s in the illustration.

2.2. Parallel reduction operators

Operators transform a binary image space; those which do so by only changing some **ones** to **zeros** (this is referred to as *reduction* or *deletion* of **ones**) are called *reduction operators,* and only these operators are considered in this paper. (Those which do so by only changing some **zeros** to **ones** are called *augmentation operators*). The *support* of a reduction operator $\mathbf{0}$ at a pixel p is the set of pixels whose values (0 or 1) are used to determine whether p is to be deleted by $\mathbf{0}$; an important property of such operators is that the support at any other pixel q is just the translation to q of the support at p. For example, the support of the operator which deletes a **one**, p, iff $N_8^*(p) \cap S$ is 8-connected and nonempty is $N_8(p)$; this is called a 3×3 *support.* When the support is confined to a region in the image space with small diameter it is referred to as a *local support.* Operators with local support are highly desirable in parallel implementations since larger supports require either higher time cost or higher interconnection complexity for obtaining the elements in the support required to compute the operator. The operators are applied over the image in a sequence of *iterations*. When a reduction operator is applied to only one pixel in an iteration, this is termed a sequential reduction operator. The term *parallel reduction operator* is used in this paper to indicate that the reduction operator is applied to the entire image space simultaneously within each iteration. (This is also called a *fully parallel reduction operator* and examples can be found in [2, 6, 8].) In this paper operators are understood to be fully parallel unless stated otherwise.

2.3. *A fully parallel thinning algorithm using reduction operators*

Typically, thinning algorithms iteratively delete *border* pixels (i.e., pixels *m*-adjacent to a **zero**) from elongated components in S until the remaining pixels form thin curves approximately along the medial axes of these connected components. Fully parallel reduction operators with 3×3 support cannot do successful thinning. As a result investigators, striving to maintain a 3×3 support restriction, have tended to use distinct 3×3 operators over subiterations, where the parallel operator is changed from iteration to iteration [5, 8, 12, 17, 18, 19, 21]. In another approach the image space is partitioned into *subfields* and a single operator is applied in parallel over each subfield (one subfield per iteration) in some sequence of the subfields [5, 13]. Early applications of the subfields methodology to parallel thinning [13] dealt effectively with connectivity preservation since neighbors of a deleted pixel could not change. This provides the same conditions which make sequential approaches so easy to analyze. Fully parallel operators with support larger than 3×3 have also been considered [2, 6, 8]. The following parallel thinning algorithm, HSCPN, is derived from the fastest thinning approach of Holt et al. [8] as modified in [7].

Algorithm HSCPN.

In successive iterations apply the following operator to all pixels in the image space.

Step 1. Find those **ones**, p, for which $N_8^*(p)\cap S$ is 4-connected and contains between three and six elements. These are called *HSCPN-deletable* **ones**.

Step 2. Delete all HSCPN-deletable **ones** except those which satisfy any of the following conditions on $N_8(p)$:

(a) $p_2=p_6=1$ and p_4 is HSCPN-deletable;

(b) $p_4=p_8=1$ and p_6 is HSCPN-deletable; or

(c) p_4, p_5, and p_6 are HSCPN-deletable,

where specific elements of $N_8(p)$ are defined as

$$\begin{array}{ccc} p_1 & p_2 & p_3 \\ p_8 & p & p_4. \\ p_7 & p_6 & p_5 \end{array}$$

The algorithm terminates when no further deletions occur in an iteration.

Step 1 defines a general "deletability" condition and Step 2 provides certain "preservation" conditions required to preserve connectivity. Satisfying conditions 2(a) and 2(b) implies the following conditions in the neighborhood of p:

$$\begin{array}{cccc} b & 1 & 1 & c \\ 0 & p & q & 0 \\ a & 1 & 1 & d \end{array} \quad \text{and} \quad \begin{array}{ccc} a & 0 & b \\ 1 & p & 1 \\ 1 & q & 1 \\ d & 0 & c \end{array},$$

respectively, where $p=q=1$ and $\{a, b\}$ and $\{c, d\}$ each contain at most one **one**. The rationale for condition 2(a) (2(b)) is to explicitly preserve certain **ones** in vertically (horizontally) oriented 2-wide rectangular 8-components. The rationale for condition 2(c) is to preserve one **one** in the 2×2 pixel 8-component. If restricted to 3×3 operators, Step 1 can be computed in parallel in one iteration; but, Step 2 uses intermediate results from Step 1 and requires a second parallel iteration. With this support restriction this algorithm uses a parallel reduction operator, but the operator alternates between execution of Steps 1 and 2 in alternate iterations. Using larger support operators both Steps 1 and 2 can be performed in parallel in one iteration where the operator has support $\langle s\rangle \cup \{p\}$,

$$\begin{array}{cccc} s & s & s & s \\ s & p & s & s \\ s & s & s & s \\ s & s & s & s \end{array},$$

since this is the support required to compute the conditions in Steps 1 and 2. In this context HSCPN is a fully parallel reduction algorithm, since the operator is unchanged from iteration to iteration. It is desirable to use the fully parallel implementation since this reduces by a factor of 2 the number of required iterations. Proofs for connectivity preservation for this algorithm have been given in [7, 8]. Figure 2 illustrates the behavior of this algorithm for simple 8-components and Fig. 1 illustrates a typical medial curve result for this algorithm on a natural image.

$$\begin{array}{cccccccccc} 1&1&1&1&1&1&1&1&1&1\\ 1&\blacksquare&\blacksquare&\blacksquare&\blacksquare&\blacksquare&\blacksquare&\blacksquare&2&1\\ 1&2&2&2&2&2&2&2&2&1\\ 1&1&1&1&1&1&1&1&1&1 \end{array} \qquad \begin{array}{cccc} 1&1&1&1\\ 1&\blacksquare&2&1\\ 1&2&2&1\\ 1&1&1&1 \end{array} \qquad \begin{array}{ccccccccccccc} \blacksquare&\blacksquare&1&1&&&&&&&&&\\ &1&\blacksquare&\blacksquare&1&&&&&&&&\\ &&1&\blacksquare&\blacksquare&1&&&&&&&\\ &&&1&\blacksquare&\blacksquare&1&&&&&&\\ &&&&1&\blacksquare&\blacksquare&1&&&&&\\ &&&&&1&\blacksquare&\blacksquare&1&&&&\\ &&&&&&1&\blacksquare&\blacksquare&1&&&\\ &&&&&&&1&\blacksquare&\blacksquare&1&&\\ &&&&&&&&1&\blacksquare&\blacksquare&1&\\ &&&&&&&&&1&1&\blacksquare&\blacksquare \end{array}$$

Fig. 2. Examples of HSCPN for simple rectangular 8-components in S. Numbers indicate the iteration when the **one** at that position is deleted. Symbol ■ denotes pixels remaining which form the medial curve estimate.

3. Classical connectivity preservation tests for reduction operators

Connectivity preservation can be characterized in many equivalent ways. In this section a characterization is used which formed the basis for early connectivity preservation proofs for parallel thinning algorithms [17, 19]. Because only reduction operators are being considered, connectivity preservation can be expressed as four key properties which must be maintained at each iteration of any parallel process which uses only reduction operators. Note that $(k, m) = (8, 4)$ for 8-4 spaces and $(6, 6)$ for 6-6 spaces. All definitions and results in Sections 3, 4 and 5 apply to both 8-4 and 6-6 spaces unless stated otherwise.

Definition 3.1. A reduction operator, **0**, is said to be *connectivity preserving* if all of the following properties are satisfied in each iteration:

C1: **0** must not transform a k-component of S into two or more distinct k-components of S;

C2: **0** must not completely delete a k-component of S;

C3: **0** must not merge distinct m-components of S' into a single m-component of S'; and

C4: **0** must not create a new m-component in S'.

There is a fundamental class of reduction operators of substantial interest when connectivity preservation is a concern; this forms the basis for most of the results in this paper.

Definition 3.2. A reduction operator, **0**, belongs to *Class-R* when a **one**, p, is deleted by **0** only if the following two conditions are satisfied:

1: p is m-adjacent to a **zero** in S' and

2: $N_k^*(p) \cap S$ is k-connected and nonempty.

Such **ones** are called *k-simple.*

It has been shown in 8-4 spaces that a connectivity preserving reduction operator with 3×3 support must belong to Class-R and sequential applications of a Class-R operator always preserve connectivity properties [16, 20]. This follows similarly in 6-6 spaces for operators with support $N_6(p)$. For Class-R operators C4 is satisfied by part 1 of Definition 3.2 and a separate test is unnecessary. Condition C4 is not considered further.

Rosenfeld [17] gave methods of proving satisfaction of C1 and C3 in parallel thinning (reduction) algorithms in 8-4 spaces which are found within Propositions 4 and 2, respectively in [17]. Aspects of his proofs for satisfying C1 and C3 are reinterpreted as connectivity preservation tests and are referred to here as RC1-k and RC3-k, respectively. The tests will be reviewed for 8-4 spaces and extended to 6-6 spaces. Test RC2-k is also defined and its role will be explained in Propositions 3.5 and 3.6.

Definition 3.3. Connectivity preservation tests for C1, C2 and C3:

RC1-k: Whenever $p \in S$ is deletable by a reduction operator **0** and $x, y \in N_k^*(p) \cap S$, then there is a k-path from x to y in $N_k^*(p) \cap S$ containing no **ones** deletable by **0** other than (perhaps) x or y.

RC2-k: No mutually k-connected component of S is completely deleted by **0**. Here, a set containing more than one element in which each pair of distinct elements are k-adjacent is called *mutually k-connected.* (These sets are illustrated in Fig. 3 and discussed below for 8-4 and 6-6 spaces.)

RC3-k: Whenever $p \in S$ is deletable by a reduction operator **0** and $x, y \in N_m^*(p)$, x in S', and y in S' or y in S but deletable by **0**, then there is an m-path from x to y in $N_k^*(p)$ containing no points in S other than (perhaps) y.

```
1      1 1      1    1    1 1    1 1      1     1      1 1
  1           1      1    1         1   1 1     1 1    1 1
```

RC2-8 (R3-8)

```
 1     1     1 1     1      1 1
1       1          1 1     1
```

RC2-6 (R3-6)

Fig. 3. Test patterns for C2 tests RC2-8 (R3-8) and RC2-6 (R3-6).

The reader can read or reconstruct the proofs in [17] and note that they hold in the 6-6 case as well which gives:

Proposition 3.4 (from [17] in the 8-4 case). (a) *Satisfying* RC1-k *implies that* C1 *is satisfied and* (b) *satisfying* RC3-k *implies that* C3 *is satisfied.*

Finally to show C2 it is sufficient to determine for any p in any S that all the **ones** in $N_k(p)$ cannot be deleted by **0** within one iteration [5, 7]. It is shown below that a simpler test for C2, i.e., RC2-k, is available given that RC1-k is satisfied also.

Proposition 3.5. *If a Class-R parallel reduction operator* **0** *satisfies* RC1-k, *then all pixels in a k-component which is completely deleted by* **0** *must be mutually k-adjacent.*

Proof. Only k-components with three or more **ones** need be considered. Arguing by contradiction assume a completely deleted k-component, Q, contains a pixel $p \in S$ and two **ones** p_1, p_2 in $N_k^*(p)$, which are not k-adjacent. In order to satisfy RC1-k after p is deleted, p_1 and p_2 must be k-connected in a nonempty k-path in $N_k^*(p)$. By RC1-k, this path is not completely deleted in the same iteration with p, giving the desired contradiction. □

In an 8-4 space there are nine mutually k-connected sets (independent of location but not orientation) each containing 2-4 elements; in a 6-6 space there are five such, with 2-3 elements. These are illustrated in Fig. 3.

Satisfaction of RC1-k for a Class-R operator also guarantees that RC3-k is satisfied and the proof is left to the reader. This and Proposition 3.5 give the following.

Proposition 3.6. *Let* $\mathbf{0}$ *be a Class-R reduction operator satisfying* RC1-k. *Then*:
(a) $\mathbf{0}$ *satisfies* RC3-k (*so* $\mathbf{0}$ *satisfies* C3 *as well as* C1),
(b) *if* $\mathbf{0}$ *also satisfies* RC2-k, *then* $\mathbf{0}$ *satisfies* C2.

Thus, for all Class-R operators satisfaction of RC1-k is sufficient to guarantee C1 and C3; and C2 is confirmed by determining that a small set of small k-components (illustrated in Fig. 3) is not completely deleted.

4. Ronse's tests and test equivalences

Ronse [15] has reported a rather simple set of sufficient conditions to determine if a parallel reduction operator, $\mathbf{0}$, preserves connectivity in 8-4 or 4-8 spaces. Below his conditions are revised using (and re-observing) some of his results [14, 15]. His tests are also extended to 6-6 spaces.

Definition 4.1. A subset D of S is *k-deletable* if the operation defined by deleting D from S is connectivity preserving (see Definition 3.1).

Notice (as Ronse did in the 8-4 case) that $\{p\}$ is k-deletable iff p is k-simple. In the following the Ronse tests are expressed and extended:

Definition 4.2. The extended Ronse tests are:
R1-k: Any **one** deleted by $\mathbf{0}$ must be k-deletable.
R2-k: If two m-adjacent **ones** p, q are deleted together by $\mathbf{0}$, where $N_k^*(\{p, q\}) \cap S$ is nonempty, then $\{p, q\}$ must be k-deletable.
R3-k: No k-component in S composed of two, three or four mutually k-adjacent **ones**, is completely deleted in one iteration by $\mathbf{0}$.

For the 8-4 instance of this test R1-8 is precisely as stated by Ronse and R2-8 uses results in [15] to simplify the complexity of his version of the R2-8 test. R3-k is identical to the C2 test, RC2-k, identified in Section 3. The test set for R3-k is illustrated in Fig. 3. An operator satisfies R1-k iff it is in Class-R. The following result is needed to facilitate computing Ronse's connectivity test and to facilitate proofs following in this section.

Proposition 4.3. *Given* $p, q \in S$ *and* p, q *m-adjacent, any two of the following three conditions imply the third*: (a) $\{p\}$ *is k-deletable*; (b) $\{q\}$ *is k-deletable after p is deleted*; *and* (c) $\{p, q\}$ *is k-deletable.*

The proof is left to the reader (see also Ronse [14, 15]).

For the Ronse R1-8 test one considers the $2^8 = 256$ distinct possible patterns of **ones** in $N_k^*(p)$ for a given **one**, p, and determines for which of these p is not 8-deletable. These are the patterns for which the algorithm under test must not delete p. There are 140 such and they are denoted R1-8 *test patterns*. For the Ronse R2-8 test (where it is assumed that R1-8 has been satisfied) one must consider test patterns for which $\{p\}$ and $\{q\}$ are both separately 8-deletable; $\{p, q\}$ is not 8-deletable; and $N_8^*(\{p, q\}) \cap S$ is nonempty, i.e., $\{p, q\}$ is not a 2-pixel component of S. For each of these the parallel reduction operator under test must not delete both p and q. Of the 2^{10} possible patterns of **ones** (for one orientation of p, q) in $N_k^*(\{p, q\})$, 192 are of this sort. Examples of such patterns are:

$$\begin{matrix} 0 & 1 & 0 & 0 \\ 0 & p & q & 0 \\ 0 & 0 & 1 & 0 \end{matrix} \qquad \begin{matrix} 1 & 1 & 0 & 0 \\ 1 & p & q & 0 \\ 0 & 0 & 1 & 1 \end{matrix} \qquad \begin{matrix} 0 & 1 & 1 & 0 \\ 0 & p & q & 0. \\ 0 & 1 & 1 & 0 \end{matrix}$$

This test set is generated for p and q oriented horizontally. If the operator under test is symmetrical for 90° rotations, then this test set is sufficient. If the operator is not symmetrical, then an expanded test set must be considered which includes the 90° rotations of this test set. This effectively doubles the number of test patterns. The Ronse R3-8 test (where it is assumed that R1-8 and R2-8 are satisfied) is performed by determining that the nine test patterns illustrated in Fig. 3 are not completely deleted by the algorithm under test. The major part of the complexity of the Ronse test is in R2-8. The R1-6, R2-6 and R3-6 tests for 6-6 spaces are defined analogously. These tests require 34, 36, and 5 test patterns, respectively. For R2-6 only one of the three possible orientations of p and q is considered; and, if the operator is not symmetrical for 60° rotations, then the R2-6 test set should be augmented with 60° rotations of the current test set which increases the test set size by a factor of 3. Note that in any case the total number of test patterns required for the 6-6 tests is considerably less than for the 8-4 tests.

Rather than directly prove that satisfying the Ronse tests implies that connectivity is preserved, equivalences are shown between RC1-k and R2-k for operators satisfying R1-k, which achieves that end.

Proposition 4.4. *If a reduction operator,* **0**, *satisfies* R1-k *and* RC1-k, *then* **0** *satisfies* R2-k, *for* $k = 6$ *or* 8.

Proof. Working by contradiction assume that R1-k and RC1-k are satisfied but R2-k is not satisfied; i.e., m-adjacent **ones** p, q are deleted by **0**, but $\{p, q\}$ is not k-deletable and $S \cap N_k^*(\{p, q\})$ is nonempty. Since **0** satisfies R1-k, $\{p\}$ is k-deletable and by Proposition 4.3, $\{q\}$ is not k-deletable after deletion of p. Also, $(N_k^*(q) - \{p\}) \cap S$ is nonempty since $\{p\}$ is k-deletable. Therefore, there exist x, y in $N_k^*(q) \cap S$ which are k-connected in $N_k^*(q)$, but are not k-connected in $N_k^*(q) - \{p\}$. But passing RC1-k implies that x, y must remain k-connected in $N_k^*(q)$ and thus p could not be deleted by **0**, which gives the desired contradiction. □

Proposition 4.5. *If a reduction operator,* **0**, *satisfies* R1-k *and* R2-k, *then* **0** *satisfies* RC1-k.

The proof uses the following well-known lemma:

Lemma 4.6. *Let u and v be distinct points in $N_k^*(x)$ that are not k-adjacent to each other. Then there are just two minimal k-paths in $N_k^*(x)$ from u to v, and each m-neighbor of x, distinct from u and v, lies on one of these two k-paths.*

Proof of Proposition 4.5. Let x be a **one** that is deleted by **0** and let u and v be distinct **ones** in $N_k^*(x)$ that are not k-adjacent to each other. Since x is deleted by **0**, and **0** satisfies R1-k, $\{x\}$ is k-deletable. So there is a k-path of **ones** in $N_k^*(x)$ from u to v. Let P be a shortest possible such k-path. To prove **0** satisfies RC1-k, it is enough to show that there is no point y in $P-\{u, v\}$ that is deleted by **0**.

Suppose such a point y exists. Note that y is m-adjacent to x. (This is plain for 6-6 spaces; for 8-4 spaces the assertion follows from the fact that the predecessor and successor of y on P cannot be 8-adjacent to each other since P is the shortest possible.) Since x and y are both deleted by **0**, and **0** satisfies R1-k and R2-k, $\{y\}$ is k-deletable and $\{x, y\}$ is k-deletable. By Proposition 4.3 $\{x\}$ must be k-deletable in the image that results from deletion of y. Hence there is a k-path of **ones** from u to v in $N_k^*(x)-\{y\}$. Let Q be a shortest possible such k-path. Since y lies on P but not on Q, Q is not the same k-path as P. So by Lemma 4.6 each m-neighbor of x lies on P or Q, and is therefore a **one**. This contradicts the fact that $\{x\}$ is k-deletable. □

Thus, for operators which satisfy R1-k (belong to Class-R), the RC1-k and R2-k tests are equivalent; and, since satisfying R1-k and RC1-k implies that C1 and C3 are satisfied, satisfying R1-k and R2-k implies that C1 and C3 are satisfied.

It is of some interest to determine if the RC1-k test could produce a simpler test implementation than R2-k. The overall RC1-k test suggests the application of 16 RC1-k local test cases for the 8-4 case and nine local test cases for the 6-6 case, as illustrated in Fig. 4, where x and y take any non-k-adjacent positions in $N_k^*(p)$. The RC1-k local test does not distinguish between x and y (which are both **ones**); thus, each test case illustrated in Fig. 4 applies to the test case where x and y are interchanged. Although there are fewer RC1-k test cases than R2-k test patterns, the complexity of an implementation of an R2-k test for each R2-k test pattern is substantially lower than that for RC1-k test cases. In fact a search for a simple RC1-k test implementation leads to the Ronse test R2-k. In the following it is assumed that the operator under test, **0**, has satisfied R1-k first, i.e., **0** belongs to Class-R. The two non-k-adjacent **ones**, x and y, in $N_k^*(p)$ must be k-connected in $N_k^*(p)$ via exactly one of two k-paths whose interior elements, A or B, are composed

$$
\begin{array}{ccc} x & \cdot & y \\ \cdot & p & \cdot \\ \cdot & \cdot & \cdot \end{array}
\quad
\begin{array}{ccc} x & \cdot & \cdot \\ \cdot & p & y \\ \cdot & \cdot & \cdot \end{array}
\quad
\begin{array}{ccc} x & \cdot & \cdot \\ \cdot & p & \cdot \\ \cdot & \cdot & y \end{array}
\quad
\begin{array}{ccc} x & \cdot & \cdot \\ \cdot & p & \cdot \\ \cdot & y & \cdot \end{array}
\quad
\begin{array}{ccc} x & \cdot & \cdot \\ \cdot & p & \cdot \\ y & \cdot & \cdot \end{array}
\quad
\begin{array}{ccc} \cdot & x & \cdot \\ \cdot & p & \cdot \\ y & \cdot & \cdot \end{array}
$$

$$
\begin{array}{ccc} \cdot & x & \cdot \\ \cdot & p & \cdot \\ \cdot & y & \cdot \end{array}
\quad
\begin{array}{ccc} \cdot & x & \cdot \\ \cdot & p & \cdot \\ \cdot & \cdot & y \end{array}
\quad
\begin{array}{ccc} \cdot & \cdot & x \\ \cdot & p & \cdot \\ \cdot & \cdot & y \end{array}
\quad
\begin{array}{ccc} \cdot & \cdot & x \\ \cdot & p & \cdot \\ \cdot & y & \cdot \end{array}
\quad
\begin{array}{ccc} \cdot & \cdot & x \\ \cdot & p & \cdot \\ y & \cdot & \cdot \end{array}
\quad
\begin{array}{ccc} \cdot & \cdot & x \\ y & p & \cdot \\ \cdot & \cdot & \cdot \end{array}
$$

$$
\begin{array}{ccc} \cdot & \cdot & \cdot \\ \cdot & p & x \\ y & \cdot & \cdot \end{array}
\quad
\begin{array}{ccc} \cdot & \cdot & \cdot \\ y & p & x \\ \cdot & \cdot & \cdot \end{array}
\quad
\begin{array}{ccc} \cdot & \cdot & \cdot \\ y & p & \cdot \\ \cdot & \cdot & x \end{array}
\quad
\begin{array}{ccc} \cdot & \cdot & \cdot \\ \cdot & p & \cdot \\ y & \cdot & x \end{array}
$$

RC1-8 test cases

$$
\begin{array}{ccccc} & x & & \cdot & \\ \cdot & & p & & y \\ & \cdot & & \cdot & \end{array}
\quad
\begin{array}{ccccc} & x & & \cdot & \\ \cdot & & p & & \cdot \\ & \cdot & & y & \end{array}
\quad
\begin{array}{ccccc} & x & & \cdot & \\ \cdot & & p & & \cdot \\ & y & & \cdot & \end{array}
\quad
\begin{array}{ccccc} & \cdot & & x & \\ \cdot & & p & & \cdot \\ & \cdot & & y & \end{array}
\quad
\begin{array}{ccccc} & \cdot & & x & \\ \cdot & & p & & \cdot \\ & y & & \cdot & \end{array}
$$

$$
\begin{array}{ccccc} & \cdot & & x & \\ y & & p & & \cdot \\ & \cdot & & \cdot & \end{array}
\quad
\begin{array}{ccccc} & \cdot & & \cdot & \\ \cdot & & p & & x \\ & y & & \cdot & \end{array}
\quad
\begin{array}{ccccc} & \cdot & & \cdot & \\ y & & p & & x \\ & \cdot & & \cdot & \end{array}
\quad
\begin{array}{ccccc} & \cdot & & \cdot & \\ y & & p & & \cdot \\ & \cdot & & x & \end{array}
$$

RC1-6 test cases

Fig. 4. RC1-k local test cases.

solely of **ones** in $N_m^*(p)$ as illustrated in the following 8-4 and 6-6 examples:

$$
\begin{array}{ccc} & a & \\ y & p & b \\ & c & x \end{array}
\qquad
\begin{array}{ccccc} & x & & a & \\ d & & p & & b \\ & c & & y & \end{array}
$$

where $A=\{a, b\}$, $B=\{c\}$ in the 8-4 example and $A=\{a, b\}$, $B=\{c, d\}$ in the 6-6 example. The other path must contain at least one **zero**, otherwise p is not k-simple. Further, to satisfy RC1-k no interior elements of the path composed of all **ones** can be deleted at iteration i when p is deleted, otherwise x and y cannot be k-connected in $N_k^*(p)$ after i. Thus, satisfying RC1-k for Class-R operators places requirements on the deletion of pairs of m-adjacent **ones** like those required by the Ronse R2-k test. Further analysis of this observation shows that R2-k test patterns can be derived in this fashion. Thus, it is likely that any test implementation for R2-k will be at least as simple as any test implementation for RC1-k.

5. RC1-k and R2-k necessity for Class-R operators

The main results to now have shown that satisfying R1-k and RC1-k or R2-k are sufficient to guarantee that C1 and C3 are satisfied. One can also demonstrate that satisfying C1 or C3 implies that R2-k (and equivalently RC1-k) is satisfied for certain restrictions on the operator support. The following result can be readily shown by checking cases.

Proposition 5.1. (a) *To verify* R2-8 *it is sufficient to check the following cases and their* 90° *rotations*:

$$
\begin{array}{cccc}
 & a & 1 & \\
b & p & q & d \\
 & c & 1 & \\
 & & & \\
 & (1) & &
\end{array}
\qquad
\begin{array}{ccc}
 & 1 & 0 \\
b & p & q \\
 & c & 1 \\
 & & \\
 & (2) &
\end{array}
\qquad
\begin{array}{ccc}
 & a & 1 \\
b & p & q \\
 & 1 & 0 \\
 & & \\
 & (3) &
\end{array}
\qquad
\begin{array}{ccc}
 & 1 & 0 \\
0 & p & q; \\
 & 1 & 0 \\
 & & \\
 & (4) &
\end{array}
$$

where: *for case* (1), $d=0$ *and* $\{a, b, c\}$ *contains at least one* **zero**, *except* $a=c=0$, $b=1$ *is not allowed*; *for case* (2), $\{b, c\}$ *contains at least one* **zero**; *and for case* (3), $\{a, b\}$ *contains at least one* **zero**. (b) *To verify* R2-6 *it is sufficient to check the following case and its* $+/-60°$ *rotations*:

$$
\begin{array}{ccccccc}
 & a & & 1 & & g & \\
b & & p & & q & & f \\
 & c & & 1 & & e & \\
 & & & & & & \\
 & & (5) & & & &
\end{array}
$$

where the sets $\{a, b, c\}$ *and* $\{g, f, e\}$ *each contain at least one* **zero**, *except neither* $a=c=0$, $b=1$ *nor* $g=e=0$, $f=1$ *is allowed.*

Any unspecified pixels in $N_k^*(\{p, q\})$ *may be either* **zero** *or* **one**.

Proposition 5.2. *A Class-R parallel reduction operator* **0** *whose support is* $N_k(p)$ *which satisfies* C1 *or* C3 *also satisfies* R2-*k*.

Proof. Assume, by way of contradiction, that R2-*k* fails. Consider the test cases of Proposition 5.1 which enumerate the cases where R2-*k* could fail. For the C1 part of the proof, note that for each of the five cases there exist two **ones**, *h* and *j*, in different *k*-components of $S \cap N_k^*(\{p, q\})$. For each of these cases a test pattern can be constructed with all **zeros** in the complement of $N_k(\{p, q\})$ except for *k*-paths in *S* containing *h* and *j*. Parts of these paths can be guaranteed not to be deletable by a Class-R operator and these parts can be guaranteed to not be *k*-connected after deletion of *p* and *q*. For example, the following test image can be constructed from case (1):

$$
\begin{array}{cccc}
 & 0 & 1 & 0 \\
 & 0 & r & 0 \\
 & a & 1 & \\
b & p & q & 0 \\
 & c & 1 & \\
 & 0 & t & 0 \\
 & 0 & 1 & 0
\end{array}
$$

where $r=t=1$; and, *r* and *t* are not deletable by **0** (i.e., *r* and *t* are not 8-simple) and are not 8-connected in *S* after deletion of *p* and *q* violating C1. (Unspecified pixels are assumed to all be **zeros**.) Similar examples can be designed for each of the five test cases in Proposition 5.1. For the C3 part of the proof note that for each of the five test cases in Proposition 5.1 there exist two **zeros**, v and w, in $N_m^*(\{p, q\})$ which are not m-connected in $N_k^*(\{p, q\}) \cap S'$ just prior to deletion of p and q, at iteration i. For each case a test image can be constructed where all m-neighbors of the **zeros** in $N_k^*(\{p, q\})$ which are in the complement of $N_k^*(\{p, q\})$ are assigned as **ones**. This guarantees that v and w are not m-connected in S' prior to i and that deletion of p and q will violate C3. This construct is illustrated for a particular instance of case (1) where $a = c = 1$ and $b = 0$:

$$\begin{array}{cccccc}
1 & 1 & & & 1 & \\
1 & 0 & 1 & 1 & 0 & 1 \\
1 & 0 & p & q & 0 & 1 \\
1 & 0 & 1 & 1 & 0 & 1 \\
 & 1 & & & 1 &
\end{array}$$

and where the two distinct components of **zeros** in $N_k^*(\{p, q\})$ will be merged after deletion of p and q violating C3. □

It is an interesting question to see how much larger the support for **0** can be before the necessity of R2-k breaks down. For C1 the support apparently can be no larger than that given in Proposition 5.2. This can be seen by considering the following two operators where failure to satisfy R2-k does not imply failure to satisfy C1. First for the 8-4 case consider the operator, $\mathbf{0}_1$, which deletes a **one** p iff either of the following neighboring relations holds:

$$\begin{array}{ccc}
 & 1 & \\
1 & 0 & 1 \\
1 & p & a \\
0 & 1 & 0
\end{array}
\quad \text{or} \quad
\begin{array}{ccc}
0 & 1 & 1 \\
a & p & 1 \\
1 & 0 & 0
\end{array}$$

where $a = 1$. In $N_8^*(p)$, a is the only pixel which could be deleted by $\mathbf{0}_1$. Now consider the image:

$$\begin{array}{cccc}
 & 1 & & \\
1 & 0 & 1 & 1 \\
1 & r & q & 1 \\
0 & 1 & 0 & 0
\end{array}$$

where unspecified pixels are all **zeros**. Pixels r and q are deletable by $\mathbf{0}_1$ and R2-8 is not satisfied. But, it can be shown that C1 is satisfied. (Note that C3 is violated and this suggests that C3 operator support requirements may differ from those for C1 in 8-4 spaces.) For the 6-6 case consider the operator, $\mathbf{0}_2$, which deletes a **one** p iff any of the following neighboring relations holds:

$$\begin{array}{ccccc}
 & & 0 & & \\
 & 0 & & 1 & \\
0 & & p & & 1 \\
 & 0 & & 1 &
\end{array},
\quad
\begin{array}{ccccc}
 & & 0 & & \\
 & 1 & & 0 & \\
1 & & p & & 0 \\
 & 1 & & 0 &
\end{array}
\quad \text{or} \quad
\begin{array}{ccccc}
 & 0 & & 0 & \\
0 & & p & & 0 \\
 & 1 & & 1 &
\end{array}.$$

Two 6-adjacent **ones** r and q are deleted by $\mathbf{0}_2$ in the following cases:

$$\begin{array}{ccccccc} & & 0 & & 0 & & \\ & 0 & & b & & 0 & \\ 0 & & r & & q & & 0, \\ & 0 & & 1 & & 0 & \\ & & & (1) & & & \end{array} \quad \begin{array}{cccccc} & & 0 & & 0 & \\ & 0 & & q & & 0 \\ 0 & & r & & 1 & \\ & 0 & & 1 & & \\ & & & (2) & & \end{array} \quad \text{or} \quad \begin{array}{cccccc} & 0 & & 0 & & \\ 0 & & r & & 0 & \\ & 1 & & q & & 0 \\ & & 1 & & 0 & \\ & & & (3) & & \end{array}$$

where $b = 1$ and is deleted by $\mathbf{0}_2$. R2-6 is not satisfied in case (1). Nevertheless, it can be shown that $\mathbf{0}_2$ satisfies C1 and C3. Through symmetries and similar examples it can be shown that when the operator support includes $N_k(p)$ plus any single additional pixel in the regions $\langle a \rangle$ below for the 8-4 or 6-6 cases

$$\begin{array}{ccccc} & a & a & a & \\ a & s & s & s & a \\ a & s & p & s & a, \\ a & s & s & s & a \\ & a & a & a & \end{array} \qquad \begin{array}{ccccccc} & & & a & & & \\ a & & s & & s & & a \\ & s & & p & & s & , \\ a & & s & & s & & a \\ & & & a & & & \end{array}$$

where p is the pixel under consideration; then an operator can be designed with this support which satisfies C1 but fails to satisfy R2-k for at least one test pattern. Further, for 6-6 instances of these cases satisfying C3 does not in general imply that R2-6 is satisfied.

Finally, we address the question as to how large the support may be before the necessity of R2-8 for satisfying C3 is lost. Consider the operator, $\mathbf{0}_3$, which deletes a **one**, p, iff any of the following neighboring conditions holds:

$$\begin{array}{ccc} & 0 & 0 \\ 0 & 1 & 0 \\ 1 & p & 0, \\ 1 & 1 & 0 \end{array} \quad \begin{array}{ccc} & 0 & 0 \\ 0 & 0 & 1 \\ 0 & p & 1 \\ 0 & 1 & 1 \end{array} \quad \text{or} \quad \begin{array}{ccc} 0 & 0 & 0 \\ 0 & p & 0. \\ 1 & 1 & 0 \end{array}$$

Two 4-adjacent **ones**, r and q, can only be deleted together by $\mathbf{0}_3$ for the following two cases:

$$\begin{array}{cccc} & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & r & q & 0, \\ 0 & 1 & 1 & 0 \\ & (1) & & \end{array} \quad \begin{array}{ccc} 0 & 0 & 0 \\ 0 & r & 0 \\ 1 & q & 0. \\ 1 & 1 & 0 \\ & (2) & \end{array}$$

R2-8 is not satisfied for case (1). But, it can be shown that the operator satisfies C3. (C1 is also satisfied since the **one** North of q is deleted by $\mathbf{0}_3$.) Thus, even two additional pixels in the operator support beyond 3×3 can remove R2-8 necessity for C3. But, there are classes of somewhat larger support which can produce this necessity.

Proposition 5.3. *A Class-R parallel reduction operator,* **0**, *defined over an* 8-4 *space with the support constraint given below and which satisfies* C3 *will also satisfy* R2-8. *The support for* **0** *at any pixel p includes p and may include members of* $\langle s\rangle$ *and at most one pixel drawn from each set* $\langle a\rangle$, $\langle b\rangle$, $\langle c\rangle$, *and* $\langle d\rangle$ *as illustrated below*:

$$\begin{array}{ccccc} & a & a & a & \\ d & s & s & s & b \\ d & s & p & s & b \\ d & s & s & s & b \\ & c & c & c & \end{array}$$

Proof. Assume by way of contradiction that R2-8 fails. Consider the 8-4 test cases given in Proposition 5.1. It is clear that for each case there are two **zeros**, z_1 and z_2, in $N_8^*(\{p, q\})$ which are not 4-connected in $N_8^*(\{p, q\})$ before deletion of p and q at iteration i. In order to avoid violation of C3 it must be argued that z_1 and z_2 are 4-connected in S' before i. To construct an example in which C3 fails, let U be the union of the supports of p and q and assign every point of U' to S. Based on the support limitation for **0** and for one orientation of p and q, U is drawn from

$$\begin{array}{cccccc} & e & e & e & e & \\ h & s & s & s & s & f \\ h & s & p & q & s & f \\ h & s & s & s & s & f \\ & g & g & g & g & \end{array}$$

where U may include p, q, any elements of $\langle s\rangle$, at most one element of $\langle f\rangle$ and $\langle h\rangle$, and at most two 4-adjacent elements of $\langle e\rangle$ and $\langle g\rangle$. For case (1) of Proposition 5.1 associate z_1 with a **zero** in $\{a, b, c\}$ and z_2 with d. In order to 4-connect z_1 and z_2 within U one must assign all of $\langle x\rangle$ or all of $\langle y\rangle$ to S' in the following instance of case (1):

$$\begin{array}{cccc} & x & x & x \\ & a & 1 & \\ b & p & q & 0. \\ & c & 1 & \\ & y & y & y \end{array}$$

But U contains at most two elements from $\langle x\rangle$ and two elements from $\langle y\rangle$. The remaining element in $\langle x\rangle$ and $\langle y\rangle$ has been assigned to S and z_1 and z_2 are not 4-connected in S' before deletion of p and q.

Thus, an example exists built from any case (1) instance for which **0** violates C3. Similar arguments give this result for the other three 8-4 cases in Proposition 5.1. □

Thus, for all Class-R operators with supports restricted as given in Proposition 5.3, R2-8 (and equivalently RC1-8) is a necessary and sufficient test for proving both C1 and C3.

6. Ronse test implementation for 8-4 spaces

The Ronse tests for the 8-4 space have been implemented in Fortran and run on a VAX cluster of three 8800 central processors under VMS 5.2. This implementation for these tests uses three precomputed files containing the required test sets. When the operator, **0**, under test has 3×3 or smaller support, **0** is applied directly to pixel p of each 3×3 test pattern in the R1-8 test file; *p and q* of each 3×4 test pattern in the R2-8 test file; and to *each* pixel in each 8-component of the R3-8 test file. For this case the three tests combined require 548 applications of **0**. When **0** has a support larger than 3×3, the R1-8 test patterns are augmented by enumerating all possible patterns in the additional support area and R2-8 test patterns are augmented similarly. This is illustrated in Fig. 5 where $\langle r \rangle$ represents the R1-8 and R2-8 test pattern regions and $\langle e \rangle$ represents additional test pattern pixels required by **0**'s support. In this example each R1-8 test pattern expands to $2^2 = 4$ test patterns and each R2-8 test pattern expands by a factor of $2^3 = 8$. The number of test patterns required for R1-8 and R2-8 rises exponentially with number of additional pixels, $|\langle e \rangle|$, in the additional support area, i.e., by a factor $2^{|\langle e \rangle|}$. R3-8 complexity is unaffected by support size. Table 1 illustrates R1-8 and R2-8 test set sizes for various **0** support sizes. As indicated in Section 4 test set sizes for R1-6 and R2-6 would be substantially lower for operators with the 7-pixel support $N_6(p)$ as compared to 3×3 support operators in 8-4 spaces. But, R1-6 and R2-6 test set sizes still increase exponentially with number of additional support pixels beyond $N_6(p)$. Execution of the tests for a typical 3×3 operator requires 1.4 sec. of CPU time while a 5×5 operator of similar complexity requires $\simeq$74 minutes of CPU time. If the operator **0** under test is not

	s				*e*				*e*	*e*		
s	*s*	*s*		*r*	*r*	*r*		*r*	*r*	*r*	*r*	
s	*p*	*s*	*s*	*r*	*p*	*r*	*e*	*r*	*p*	*q*	*r*	*e*
s	*s*	*s*		*r*	*r*	*r*		*r*	*r*	*r*	*r*	
0 support				R1-8				R2-8				

Fig. 5. Additional test pattern pixels, $\langle e \rangle$, implied by operator support, $\langle s \rangle \cup \{p\}$, which is larger than 3×3. The R1-8 test patterns are formed in $\langle r \rangle \cup \{p\}$ and the R2-8 test patterns are formed in $\langle r \rangle \cup \{p\} \cup \{q\}$.

Table 1. R1-8 and R2-8 test set sizes for varying **0** support

	Number of Test patterns	
0 Support	R1-8	R2-8
3×3	140	192
3×4[1]	1120	1536
4×4	17920	49152
5×5	9175040	50331648

[1] Support's long axis and orientation of $\{x_1, x_2\}$ are horizontal.

symmetrical under 90° rotations, additional tests are required for R2-8, since horizontally and vertically oriented test patterns must be considered. In this case the R2-8 test is replicated with vertical orientation and the time complexity of the R2-8 test is doubled.

Subfields approaches have been useful in thinning and shrinking [4, 5, 13]. In these approaches a partition is applied to the image space and in each iteration a parallel operator is applied only over one subset of the partition. The "checkerboard" two subfields partition:

$$\begin{matrix} 1 & 2 & 1 & 2\ldots \\ 2 & 1 & 2 & 1\ldots \\ 1 & 2 & 1 & 2\ldots \\ \vdots & \vdots & \vdots & \vdots \end{matrix}$$

has produced thinning [5] and shrinking algorithms [4] with the best known average parallel computation times (lowest numbers of iterations) when using operators with 3×3 or smaller support. A parallel two subfields reduction operator would be applied to pixels in alternate subfields in alternate iterations. It is obvious that R2-8 automatically holds for any such algorithm since 4-adjacent neighbors of any pixel are not changing in any one iteration. Thus, for these cases only R1-8 and R3-8 need to be checked. Also, the only 8-components in the R3-8 test set which could possibly be completely deleted are the 2-pixel 8-components where the **ones** are not 4-adjacent, which would allow for a simpler R3-8 test set in subfields cases. The current computer implementation allows for an arbitrary but regular subfields definition which affects only the R2-8 test.

7. Conclusions

Classical connectivity preservation tests derived mainly from Rosenfeld's connectivity preservation proofs [17] for 8-4 spaces have been reviewed and characterized for 8-4 rectangular and 6-6 hexagonal spaces and simplified tests (RC1-k, RC2-k and RC3-k) have been derived. The Ronse [15] connectivity preservation tests for 8-4 spaces have been extended to 6-6 spaces and efficient versions of these tests have been developed. Implementation complexities for the extended Ronse tests have been evaluated and it is found that test complexities for 8-4 spaces are substantially greater than those for 6-6 hexagonal spaces. For example, the number of test patterns for symmetrical operators for the 8-4 case is 341 while for 6-6 only 75 are required. It has been shown that the two test forms are essentially equivalent leading to a relatively easy and straightforward proof of the extended Ronse tests. Further, the necessity (i.e., satisfying connectivity properties implies satisfaction of the test for the property) of the tests for C1 and C3 connectivity properties has been explored and it has been shown under certain support restrictions that these tests are necessary and sufficient. A computer implementation has been described for the

Ronse tests for 8-4 spaces and specific execution times are given illustrating the practicability of the test for reduction operators with local support (i.e., 5×5 or smaller).

Acknowledgement

The author enjoyed particularly stimulating interaction with T.Y. Kong in the development of the counterexamples in Section 5 and received very thoughtful and detailed comments from the editors of the special issue, T.Y. Kong, R. Kopperman, and P. Meyer, and from the reviewers, all of which constituted a substantial contribution to this work.

References

[1] N.G. Bourbakis, A parallel-symmetric thinning algorithm, Pattern Recognition 22 (1989) 387-396.
[2] R.T. Chin, H-K. Wan, D.L. Stover and R.D. Iverson, A one-pass thinning algorithm and its parallel implementation, Comput. Vision Graphics Image Process. 40 (1987) 30-40.
[3] A.L. DeCegama, The Technology of Parallel Processing: Parallel Processing Architectures and VLSI Hardware Vol. 1 (Prentice Hall, Englewood Cliffs, NJ, 1989).
[4] M. Gökmen and R.W. Hall, Parallel shrinking algorithms using 2-subfields approaches, Comput. Vision Graphics Image Process. 52 (1990) 191-209.
[5] Z. Guo and R.W. Hall, Parallel thinning with two-subiteration algorithms, Comm. ACM 32 (1989) 359-373.
[6] Z. Guo and R.W. Hall, Fast fully parallel thinning algorithms, CVGIP: Image Understanding 55 (1992) 317-328.
[7] R.W. Hall, Fast parallel thinning algorithms: parallel speed and connectivity preservation, Comm. ACM 32 (1989) 124-131.
[8] C.M. Holt, A. Stewart, M. Clint and R.H. Perrott, An improved parallel thinning algorithm, Comm. ACM 30 (1987) 156-160.
[9] C.V. Kameswara Rao, D.E. Danielsson and B. Kruse, Checking connectivity preservation properties of some types of picture processing operations, Comput. Graphics Image Process. 8 (1978) 299-309.
[10] J. Kittler and M.J.B. Duff, Image Processing System Architectures (Wiley, New York, 1985).
[11] T.Y. Kong and A. Rosenfeld, Digital topology: Introduction and survey, Comput. Vision Graphics Image Process. 48 (1989) 357-393.
[12] H.E. Lü and P.S.P. Wang, A comment on "A fast parallel algorithm for thinning digital patterns", Comm. ACM 29 (1986) 239-242.
[13] K. Preston and M.J.B. Duff, Modern Cellular Automata (Plenum, New York, 1984).
[14] C. Ronse, A topological characterization of thinning, Theoret. Comput. Sci. 43 (1986) 31-41.
[15] C. Ronse, Minimal test patterns for connectivity preservation in parallel thinning algorithms for binary digital images, Discrete Appl. Math. 21 (1988) 67-79.
[16] A. Rosenfeld, Connectivity in digital pictures, J. ACM 17 (1970) 146-160.
[17] A. Rosenfeld, A characterization of parallel thinning algorithms, Inform. and Control 29 (1975) 286-291.
[18] A. Rosenfeld and A.C. Kak, Digital Picture Processing Vol. 2 (Academic Press, New York, 1982).
[19] R. Stefanelli and A. Rosenfeld, Some parallel thinning algorithms for digital pictures, J. ACM 18 (1971) 255-264.
[20] S. Yokoi, J. Toriwaki and T. Fukumura, An analysis of topological properties of digitized binary pictures using local features, Comput. Graphics Image Process. 4 (1975) 63-73.
[21] T.Y. Zhang and C.Y. Suen, A fast thinning algorithm for thinning digital patterns, Comm. ACM 27 (1984) 236-239.